Modeling and Simulating Human Activity

Maarten Sierhuis
RIACS/USRA
NASA/Ames Research Center
Mail Stop 269-3
Moffett Field, CA 90435
USA
(+01) 650-604-4917
msierhuis@mail.arc.nasa.gov

William J. Clancey
NASA/Ames Research Center
Computational Sciences
Mail Stop 269-3
Moffett Field, CA 90435
USA
(+01) 650-604-2526
bclancey@mail.arc.nasa.gov

Ron van Hoof
Caelum Research
NASA/Ames Research Center
Mail Stop 269-3
Moffett Field, CA 90435
USA
(+01) 203-531-4741
rvanhoof@mail.arc.nasa.gov

Robert de Hoog
University of Amsterdam
Department of Social Science Informatics
Roeterstraat 15
1018 WB Amsterdam
The Netherlands
(+31) 20-525-6794
dehoog@swi.psy.uva.nl

ABSTRACT

This paper describes a multi-agent approach for modeling and simulating the activities of humans and systems in organizations we refer to as work practice modeling. We describe a simulation experiment of the work practice of the Apollo 12 astronauts during the ALSEP offload activity.

Keywords

multi-agent, modeling, simulation, intelligent agents, work practice, activity

1. Introduction

In this paper, we report on the results of an experiment using a new approach for modeling and simulating the activities of humans and systems in organizations. The understanding of work processes and workflow has become more and more an integral part in the development of support technologies for organizations. However, the common modeling approaches used for modeling work processes and workflow lack, in a fundamental way, the ability to include the intricacies of the actual practices of people and systems. In the Work Systems Design and Evaluation group at NASA Ames Research Center, we are developing theories and tools for modeling the practices in a work system. In particular, we have developed a new multi-agent modeling and simulation language and environment—Brahms[1]—to study these theories [1].

2. Goals and Objectives

The goal of the experiment was to investigate the use of the Brahms-language in order to *describe an existing work practice*. The challenge we faced in this experiment was to investigate if our theory of modeling work practice, as implemented in the Brahms language, would be sufficient to describe the work practice in the chosen domain. The objectives of this first experiment were:

? Being able to represent the people, things, and places relevant to the domain.

? Represent the actual behavior of the people, second by second, over time.

? Show which of the tools and artifacts are used when, and by whom to perform certain activities.

? Include the communication between co-located and distributed people, as well as the communication tools used, and the effects of these communication tools on the practice.

The domain we chose for this experiment, and we will describe in this paper, is the work practice of the Apollo 12 astronauts in the deployment of the Apollo Lunar Surface Experiments Package (ALSEP) on the Moon. The reasons for choosing this domain are the following:

1. The work performed by the astronauts requires unique and highly skilled individuals. The complexity of the work to be described is high enough to argue that if we can model this type of work practice within Brahms, we can model most other work practices as well.
2. The ALSEP deployment work is highly distributed over the people involved, and is collaborative in nature.
3. There is no work product “flowing” through the work process. This means that this type of work is not easily represented in a workflow model. Being able to model this type of work in Brahms supports our argument for developing Brahms.
4. In order to develop a descriptive model of an existing work practice, we need to have access to a significant amount of data about the actual work practice. In practice (no pun intended), this often means a long observational and/or ethnographical study of the participants. This takes an enormous amount of effort and is a grounded research process in and of itself. However, the Apollo project has been well documented by NASA and numerous institutions, and writers [2] [3] [4] [5]. Specifically, there is a significant library of video and audiotapes taken during the actual missions [6]. This allows us to develop, verify and validate our models using independent data from the real events.

3. Theory of Modeling Work Practice

We briefly describe our theory of modeling work practice. Representing how people do work can be done at many different levels. In the knowledge engineering and AI world, people’s work has been described in terms of their problem-solving

1 Brahms is originally an acronym for "Business Redesign Agent-based Holistic Modeling System", but is now used as an internal product name for the modeling language, as well as the set of tools that comprises the product.

expertise. The theory is that we can model people's problem-solving behavior by representing this behavior in a computational model that is able to duplicate some of this behavior. Work process models, such as Petri-Net models of a work process, describe what tasks are performed and when. In workflow models we describe how a specific product "flows" through an organization's work process. This describes the sequential tasks in the work process that "touch" a work-product. All these modeling approaches describe the work in an organization at a certain level of detail. However, what is missing from all these types of modeling approaches is a representation of how work gets done. What is missing is a description of the work at the *work practice level*.

Work practice includes those aspects of the work process that make people behave a certain way in a specific situation, and at a specific moment in time. To describe people's situation-specific behavior we need to include those aspects of the situation that explain the influence on the *activity behavior* of individuals (in contrast with problem-solving behavior). Following is a brief description of the important aspects that determine an individual's situation-specific behavior.

3.1 Activity Behavior

People's behaviors are determined by the "execution" of specific activities at certain moments. This means that a person or system cannot be "alive" without being in some kind of activity. Even "doing nothing" is described in terms of a "do-nothing" or idle activity. Furthermore, what activity is being performed depends on the situational context that a person or system is in. Agents' behaviors are organized into reactive- as well as deliberative activities, inherited from groups to which agents belong. Most importantly, activities locate behaviors of people and their tools in time and space, so that resource availability and informal human participation can be taken into account [7].

Activities can be subsumed by other activities in a hierarchical structure [8] [9]. With this we mean that a person can be in multiple activities at once. For example, we can be in the activity of reading a book, while at the same time be in the higher level activity of a being on a business trip. When the phone rings in our hotel room, we get up and walk over to pick up the phone. This means that we interrupt the activity of reading our book, and start the activity of answering the phone. In a sense, we actually never stop being in the activity of reading our book, but we suspend the activity to focus on a new activity, continuing with the suspended activity when the phone call is over.

A model of activities doesn't necessarily describe the intricate details of reasoning or calculation, but instead captures aspects of the social-physical context, including space and time in which reasoning occurs [10] [11].

3.2 Context

People act based on the situation they are in. With this we mean that people behave based on their beliefs about what they experience (infer or detect) their context to be. Therefore, different people can/will have different beliefs about a similar context. If we want to model work practice, we need to be able to separate the context from people's different interpretation of that context. In order to do so, we describe context in terms of objects and artifacts that people observe and use within their environment [12]. We also describe the geographical locations of people and artifacts [13]. What describes a context is known as world-facts or simply facts. Facts represent factual information about the three-dimensional world people live in. People do not automatically have "knowledge" about those facts, and if people have "knowledge" about those facts it might not be correct [14]. For example, you can believe that your car is parked in the garage, whereas in reality someone has taken the car to go out. So, the fact is that the location of the car is wherever it has been taken, while you believe that the location of the car is the garage. You will have that belief until either someone tells you about the actual location (or wrong location) of the car, or until you go to the garage and observe (i.e. detect) that the car is not there. Of course, if the car returns before any of this takes place you will never know the car had been gone. In other words, although facts are global (the car can only be in one location), not every person can get "access" (i.e. get a belief) about that fact. Implicit in the above example is the fact that people and objects are always located. Moving from one location to another is an activity that takes time.

3.3 Communication

In order for two or more people to collaborate they need to communicate. In the Speech Act theory of Searle the meaning and intent of certain speech acts are formalized [15]. Searle describes people's action in terms of sending and receiving speech acts triggering response actions. Searle went as far as defining a taxonomy of types of speech acts in which he classified all types as embodying one of five illocutionary points (assertives, directives, commissives, expressives, declarations) [16]. Speech Act theory analyzes communication in terms of its illocutionary point, -force and propositional content. Using this type of communication analysis, we can model the sequence of (communication) actions in a collaboration activity between sender and receiver, as well as the intention and meaning of the speech act. However, in analyzing the way collaboration occurs in practice, we also need to analyze communication in terms of how it actually happens in the real world, thereby modeling collaboration as it really occurs. Speech Act theory analyzes communication in terms of patterns of commitment entered into by the speaker and the hearer. While this is important, it doesn't, for instance, take into account that a communication activity between two people works or does not work due to the communication tools used in the situated speech act. Today, communication is more and more efficient and certain communication tools are used globally. Phones, voice mail, e-mail, and fax, are communication tools that are more and more taken for granted in the way that we use them. However, it should not be taken for granted that we all have created our own practice around the use of these tools in certain situations.

This emphasizes the point that collaboration is very much defined by our practice surrounding our communication tools, and that we therefore, need to include the use of communication tools in modeling how people actually coordinate their collaboration in the real world. We need to include a model of the workings of communication tools, and how they are used in practice.

3.4 Communities of Practice

In order to describe how two different persons can perform different activities based on the same situational context, we borrow the term community of practice (CoP) from the social sciences [17]. People belong to many different communities. One way we can distinguish one community from another is in the way they are able to perform certain activities. For instance, at NASA we can distinguish the community of Apollo astronauts from the rest of the communities at NASA. We can describe the work of a particular community as a separate "group." Members of groups can perform the group's activities. Thus, we can describe people's behavior in terms of the groups they belong to.

In the next section, we will first describe how agents are modeled in Brahms. After that, we will describe the Apollo 12 ALSEP Offload domain to give the reader some background knowledge.

4. Brahms Agents

An agent is a construct that generally represents a person or robot within a workplace or other setting being modeled. Agents have a name and a location. To specify what an agent does the modeler defines activities and workframes for the agent. The key properties of agents are group membership, beliefs, activities, workframes, thoughtframes, and location. The simulation engine schedules the constrained activities of agents.

A group can represent one or more agents, either as direct members or as members of subgroups. Typically, a modeler would associate descriptions of activities with groups, so that a group represents a collection of agents that perform similar work and have similar beliefs. Depending on the purpose of the model, agents in a model may represent particular people, types of people, or pastiches. Each agent and group can be a member of any number of groups, providing that no cyclic membership results.

4.1 Elements of an Agent

A Brahms agent may have the following elements:

? *name*: The name of an agent is its unique identifier. Normally we give agents fictitious names to identify specific individuals in an organization without identifying them.

? *group-membership*: An agent can be a member of one or more groups. When an agent is a member of a group the agent will inherit attributes, relations, initial-beliefs, initial-facts, activities, workframes and thoughtframes from the group(s) it is a member of.

? *cost and time*: The cost per unit ("Cost/unit"), and the unit time for which the cost is entered ("Unit (seconds)"). Using these attributes the simulation engine can calculate cost statistics of a work process, based on a calculation of the summation of an agent's activity time.

? *location*: An agent has an initial location within the geography.

? *initial-beliefs*: A belief is a first-order predicate statement about the world [18] [19]. Beliefs are always local to an agent or object. This allows us to represent how a specific agent 'views' the state of the world [20]. Agents act based on their beliefs. Beliefs are the 'triggers' of agent's actions [21]. Initial beliefs define the initial state for an agent. Initial beliefs are turned into actual beliefs for the agent/object when the model is initialized for execution.

? *initial-facts*: Facts represent the state of the world. A fact is a first-order predicate statement about the world. Facts are, in contrast to beliefs, global. Any agent can detect a fact in the world and turn it into a belief and act on it. Objects on the other hand, react to facts without turning them into beliefs first. Initial facts define the initial state of the world. Initial facts are turned into facts in the world when the model is initialized for a simulation run. There is a fundamental difference between the "ownership" of a belief and a fact. A belief is "owned" by a specific agent during the execution of the model. No other entity in the model can access that belief without some interaction with the agent (direct or indirect). However, although initial-facts are defined with an agent or object, at execution time a fact is not "owned" by that agent or object. A fact is global, and can be acted on (in the case of objects) or detected (in the case of agents).

? *activities*: In this element the activities an agent can be engaged in are defined. Activities in Brahms take a certain amount of time, either derived or defined. An activity can have a fixed duration, or random duration based on a given time-interval. Even though an activity has a pre-specified duration (fixed or random), the actual duration of an activity depends on the context in which the agent performs the activity. An activity can be interrupted or impassed based on the detection of facts in the world, communication, or reasoning. There are a number of types of activities that are defined for the Brahms language (primitive, communicate, move, create-object, composite) Activities are executed by workframes.

? *workframes*: Workframes are rule-like constructs with preconditions constraining the execution of activities for an agent. The preconditions in a workframe are matched against the belief-set of the agent. The body of a workframe can contain consequences and activities. Consequences create new beliefs and/or facts in the world. The creation of beliefs and facts can be controlled with certainty factors. The body of a workframe is executed sequentially. Workframes can be interrupted, which means that the workframe execution is suspended, and its context saved. At continuation of the workframe the context is restored and execution continues where it was interrupted. The execution of workframes is also controlled by its priority. The available workframe (i.e. all preconditions match beliefs of the agent) with the highest priority is the current workframe being executed.

? *thoughtframes*: Thoughtframes are forward-chaining production-rules. Thoughtframes are different from workframes in that they cannot contain any activities, and therefore do not take any time. Thoughtframes can only create new beliefs, and are thus used to model reasoning behavior of the agent.

4.2 Reactive Behavior

To model humans, we need to allow for both deliberative as well as reactive behavior. Brahms combines both of these types of behaviors. Deliberative behavior of an agent is modeled using a combination of workframes and thoughtframes, as described in the previous section. Reactive behavior of agents is modeled through a construct called a *detectable*. A detectable is a mechanism by which, whenever a particular fact occurs in the world, an agent may notice it. The noticing of the fact may cause the agent to stop or to finish the activities in a workframe.

Two things can occur in a detectable. First, the agent detects the fact and the fact becomes a belief of the agent. Second, the beliefs of the agent are matched with the condition used in the detectable, and if there is a match the then-part of the detectable is executed, which may abort or interrupt the workframe. These two steps are independent: Whether or not the fact is present in the world, the condition in the second step is tested. For example, if "the color of the telephone1 is blue" is a fact and a workframe contains the following detectable condition, "the color of the telephone1 is red," an agent will obtain the belief "the color of telephone1 is blue". In the second step, "red" would be compared with "blue" and fail, so the then-part of the detectable would not be executed.

The action or then-part of a detectable defines the detectable type and is one of five keywords: continue, abort, complete, impasse, and end-activity. Detectables can be used to model reactive behavior, as well as impasses. A common example of an impasse is inaccurate or missing information. Workframes may be written to handle impasses. For example, if a supervisor is ready for a technician but does not know the technician's name, another workframe may lead the supervisor to look up the name. Impasses may also be handled by modeling social knowledge.

With a detectable, an agent may notice passive observables, as when someone shouts, a fax machine beeps, or an agent is present vying for attention. Passive observables fall into two general classes: sounds and visual states. Objects that cause a sound—fax, phone, initial bid by a person for conversation–– create the fact of the sound, which can be detected. Sounds may persist over many clock-ticks. Propagation into the surrounding space will recur as long as the object is making a sound. Propagation may be affected by geography.

5. Apollo 12 and the ALSEP Offload

One of the biggest objectives of the Apollo 12 mission was to deploy the Apollo Lunar Surface Experiments Package (ALSEP). It would be the first time the ALSEP would be deployed on the moon. The earlier Apollo 11 mission only deployed a preliminary version, called the EASEP (Early Apollo Surface Experiment Package). The ALSEP consisted of a number of independent scientific instruments that were to be deployed on the moon. The instruments were data collection devices for different scientific experiments about the moon's internal and external environment. By deploying similar ALSEP instruments over multiple Apollo missions (A12, 14, 15, 16 and 17), the ALSEP deployments created an array of data gathering instruments at different locations on the lunar surface.

To deploy the ALSEP on the lunar surface, the astronauts had to accomplish three high-level tasks. First, they had to *offload* the ALSEP from the Lunar Module (LM). Second, they had to *traverse* with the ALSEP packages to the deployment area, away from the LM. Third, they had to *deploy* each ALSEP instrument onto the surface. In this paper, we discuss the development of a work practice model for the first task, the *ALSEP Offload*.

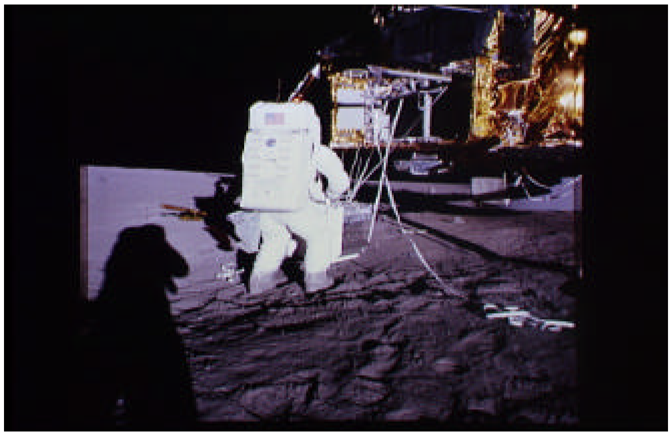


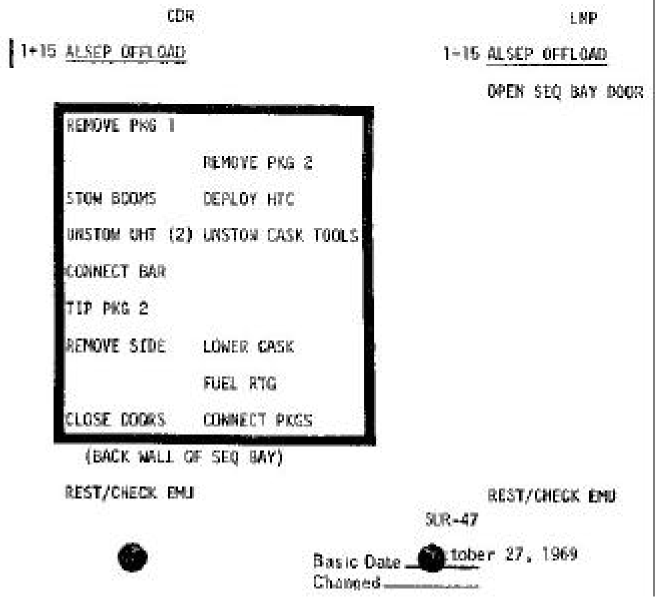
Figure 1. CDR offloading ALSEP Package 2

All the ALSEP instruments and tools used for deployment, were stored on two sub-pallets ("packages") in the Scientific Equipment Bay (SEQ Bay) during flight. Figure 1 shows Commander Pete Conrad offloading the second ALSEP package from the SEQ Bay.

The offload consisted of a number of specified (sub)activities that were trained extensively and assigned to each of the astronauts. The order in which these tasks were to be performed, and whether the Lunar Module Pilot or the Commander was to perform the task, i.e. the plan, was the same for all five missions. Figure 2 shows the plan and start-time for the Apollo 12 ALSEP Offload.

In other words, offloading the ALSEP was a highly choreographed collaborative activity performed by two astronauts working in parallel.

However, even though this high-level task was planned and choreographed up front, the plan did not include the situational variations, the actual communication and collaborative activities between the astronauts, and the communication between and coordination of activities by the Manned Spaceflight Center (MSC) in Houston. MSC, also known as Mission Control, kept track of where the astronauts were on the plan, solving unplanned problems, and monitoring and communicating life support status for the astronauts. Central in this collaborative activity is the person who played the role of Capsule Communicator (CapCom). The CapCom was the "voice" of Houston and the only person in direct communication with the astronauts. This communication happened through the voice-loop.


```
                CDR                                         LMP

1+15 ALSEP OFFLOAD                            1-15 ALSEP OFFLOAD

                                                  OPEN SEQ BAY DOOR

   REMOVE PKG 1

                   REMOVE PKG 2

   STOW BOOMS      DEPLOY HTC

   UNSTOW UHT (2)  UNSTOW CASK TOOLS

   CONNECT BAR

   TIP PKG 2

   REMOVE SIDE     LOWER CASK

                   FUEL RTG

   CLOSE DOORS     CONNECT PKGS

     (BACK WALL OF SEQ BAY)

   REST/CHECK EMU                                 REST/CHECK EMU
                                       SUR-47

                            Basic Date [illegible]tober 27, 1969
                            Changed
```
Figure 2. Apollo 12 Checklist for the ALSEP Offload

The work practice of the ALSEP Offload, or any work practice for that matter, consists of more than the sequence and distribution of tasks. As we discussed in the previous section, what constitutes the practice of the ALSEP Offload is the way the actual plan is carried out. The situational activities of the collaborators, the way they react to their environment, the way they communicate, what is said, the way they "know" how to do their tasks given the situation. It is situated action [22]. A choreographed play "executed" during the performance, planned and trained, but always different.

In the next sections, we will describe how the ALSEP Offload work is modeled in a model of work practice. The model is not a model of the problem-solving knowledge of each individual involved in this task. Instead, it is a model of the behavior of the individuals. It describes how the collaboration, coordination, and communication between the three individuals happen, and make this a fluent event. The activities of one individual are like the movements of a musician in a symphony orchestra. The communication between individuals is like the interleaved notes that seem to "tell" each musician what to play next. The artifacts and tools are like the instruments of the musician. The environment of the Moon and Mission Control is like the symphony hall. The Brahms "symphony" that is being played is planned and scored on a piece of paper (i.e. the astronaut's checklist). The orchestra has trained the piece many times (i.e. the astronaut training on Earth). However, what comes out in the performance is due to their practice, the concert hall (i.e. the Moon), and the way they play together that specific evening (i.e. EVA 1 on Apollo 12).

6. Agent Model

One of the most relevant design issues for any Brahms model is the design of the agents and the groups they belong to. The Agent Model describes to which groups the agents belong and how these groups are related to each other.

Designing an Agent Model is similar to the design of an Object Model in object-oriented design [23]. Just as the class-hierarchy in an Object Model, we need to design the group-hierarchy in the Agent Model. As a rule of thumb, we identify the communities of practice of which the agents in the model are members, and abstract them to a common denominator for all agents. All agents are members of this abstract group. The most abstract group is called the Base Group. This group exists in the Brahms Base library. It contains all attribute and relation definitions that are needed by default, such as the name of the agent, the group membership relation, and the location of the agent. We specialize this group, until we have identified all the similarities and differences between the agents. It should be noted that groups and agents could be members of multiple groups.

Figure 3 shows the Agent Model design. We start with defining our agents. Each agent represents a person in our domain, e.g. Ed Gibson, Pete Conrad, Al Bean, and Dick Gordon. We generalize the community all four agents belong to as the group of ApolloAstronauts.

We represent the role of each of the astronauts as a group. This way we can represent role specific attributes and activities at the group level. The AlsepOffloadGroup is a functional group in the sense that it doesn't specify a specific role, but a task of the agent. This group represents all work activities and attributes that have to do with the ALSEP Offload task in one group. This way, the group represents the community of agents that can perform the ALSEP Offload task. Figure 4 shows the Brahms source code of the group and agent definitions shown in Figure 3.

7. Object Model

After the Agent Model, the next model that needs to be designed is the Object Model. In this model we design the class-hierarchy of all the domain objects. Figure 5 shows the Object Model design for the Apollo 12 domain objects and artifacts. As with the Agent Model, the root-class of the class hierarchy is the class *BaseClass*. All other classes and objects inherit from this BaseClass class.

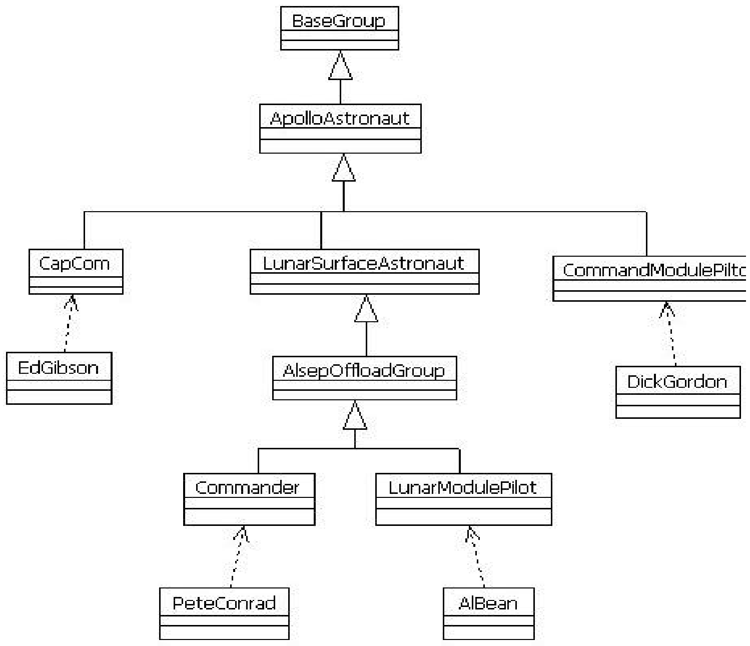


Figure 3. Apollo Agent Model Design

```
// Groups
group BaseGroup { … }
group ApolloAstronaut memberof BaseGroup { … }
group CapCom memberof ApolloAstronaut { … }
group LunarSurfaceAstronaut memberof ApolloAstronaut { … }
group CommandModulePilot memberof ApolloAstronaut { … }
group AlsepOffloadGroup memberof LunarSurfaceAstronaut { … }
group Commander memberof AlsepOffloadGroup { … }
group LunarModulePilot memberof AlsepOffloadGroup { … }

// Agents
agent PeteConrad memberof Commander { … }
agent AlBean memberof LunarModulePilot { … }
agent DickGordon memberof CommandModulePilot { … }
agent EdGibson memberof CapCom { … }
```

Figure 4. Brahms Source Code of the Agent Model

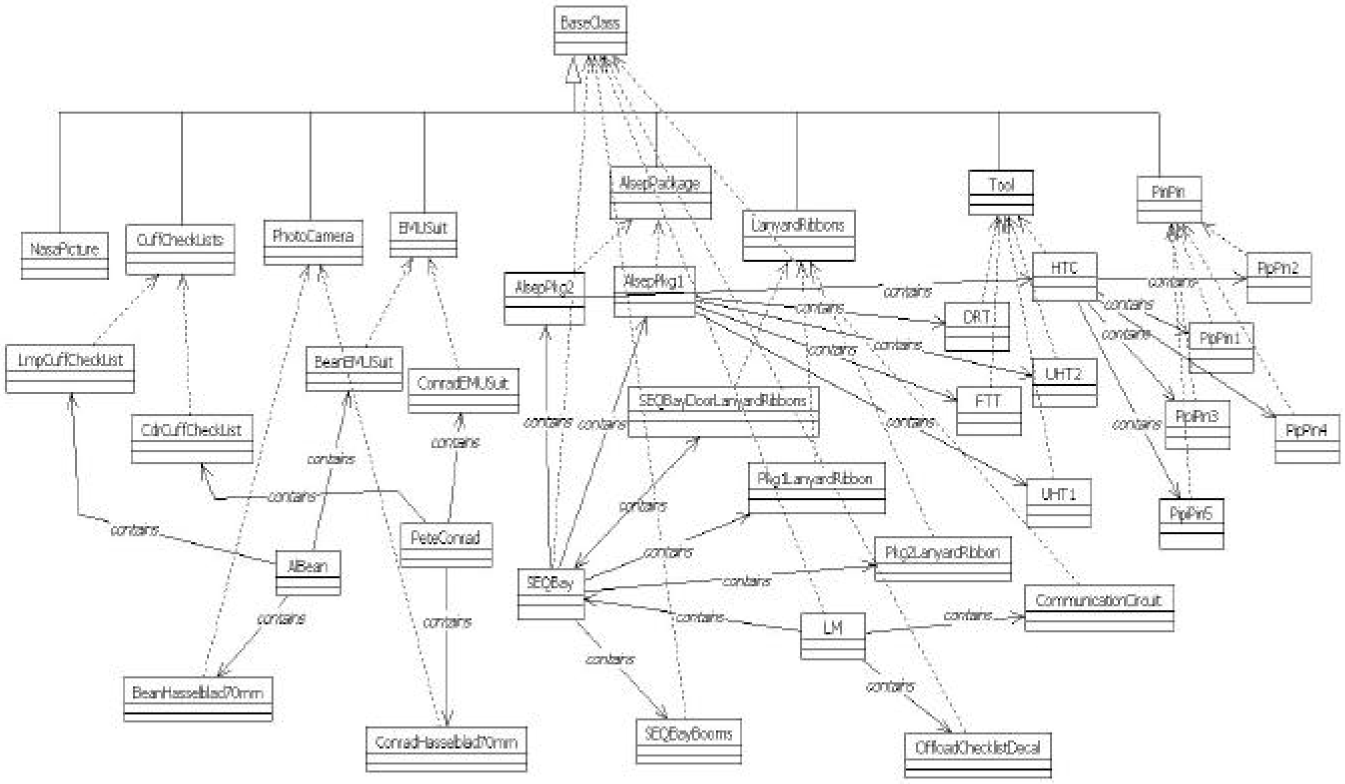


Figure 5. Apollo Object Model Design

Figure 6 shows the Brahms model source code for the LM and SEQBay objects. Both the LM and SEQBay objects are instances of BaseClass.

Besides representing the corresponding artifacts on the Apollo 12 mission, the source code also specifies the initial location of the object within the Geography Model (see next section). Both objects are located in the SEQBayArea area. Furthermore, the objects declare the attributes with which we describe the different aspects of these objects. Although we could describe any number of aspects of an object, such as the color, height, etc, we only declare those attributes that are relevant. To model the fact that the astronauts inspect the LM and the SEQ Bay's exterior appearance after the landing, we declare the attribute *exteriorAppearance* as a type symbol attribute. Using this attribute we can represent the state of the exterior of these objects [12].

Both the LM and the SEQBay objects have a fact describing the state of their exterior appearance after the landing on the moon as an initial fact for the simulation, e.g.

*(current.**exteriorAppearance** = LooksGood)*

```
// Apollo 12 objects
object LM instanceof BaseClass {
        display: "Intrepid";
        location: SEQBayArea;
        attributes:
                public symbol exteriorAppearance;
```

```
        initial_facts:
                (current.exteriorAppearance = LooksGood);
                (current contains SEQBay);
}

object SEQBay instanceof BaseClass {
        location: SEQBayArea;
        attributes:
                public symbol door;
                public symbol exteriorAppearance;
        initial_facts:
                (current.exteriorAppearance = LooksGood);
                (current.door = closed);
                (current contains AlsepPkg1);
                (current contains AlsepPkg2);
                (current contains OffloadChecklistDecal);
                (current contains SEQBayDoorLanyardRibbons);
                (current contains Pkg1LanyardRibbons);
                (current contains Pkg2LanyardRibbons);
                (current contains SEQBayBooms);
}
```

Figure 6 Apollo 12 LM and SEQ Bay Brahms objects

The status of the door of the SEQBay is modeled with the *door* attribute of type symbol. The door is in the initial state (i.e. an initial fact) of being closed, e.g.

*(current.**door** = closed)*

This represents the door of the SEQ Bay being closed at the start of the ALSEP offload. Next, we model the objects that are located within the LM and SEQ Bay. This is represented with the contains relation (see Figure 5). This relation is declared in the BaseClass class, and inherited by the LM and SEQBay objects. The fact that the SEQBay is located on the outside of the LM is represented as an initial fact in the LM object, i.e.

*(current **contains** SEQBay).*

Now that the agents and artifacts are represented, the next section describes the geography model in which the agents and artifacts are located during the simulation.

8. Geography Model

In Brahms we model geographical locations using two concepts, area-definitions and areas. Area-definitions are user-defined types of areas. Areas are instances of area-definitions. Thus an area is an instance of a specific location in the real world that is being modeled. Furthermore, areas can be part-of other areas. With this representation scheme we can represent any location at any level of detail.

For the Apollo 12 ALSEP Offload activity, the following locations are important; Earth, the Manned Spaceflight Center[2] (MSC), the Moon, the Apollo 12 landing-site ("Surveyor Crater"), the area where the SEQ Bay is located, the ALSEP deployment area, an area away from the SEQ Bay to place artifacts after offloading, and last, the lunar orbit and the Command Module ("Yankee Clipper"). Figure 7 shows the geography model design.

Figure 8 shows the Brahms source code of the area definitions (areadef) and area objects (area). The *area definition* types used to represent the area-instances are World, City and Building. It does not seem logical to give the area-definitions the names "World", "City", and "Building," and indeed it is not. The reason for this is the limitation of the current Brahms simulation engine[3].

8.1 Initial Locations

Each agent and object has an initial location in one of the lowest-level areas, (CommandModule, AwayFromTheSEQBayArea, AlsepDeploymentArea, LandingSite, SEQBayArea, or MissionControlCenter). Initial locations are locations in which an agent or object is placed during the initialization phase of the simulation. This way each agent and object starts out being located in a geographical location (an area). To define an initial location for an agent or object the modeler uses the location attribute (see Figure 6).

8.2 Movement

Agents and objects can move from one location to another. Moving from one location to another removes the agent from the starting location and moves the agent to the new location. This is accomplished by having the agent perform a move-type activity. The time the activity is active (i.e. the activity duration-time) determines how long it takes the agent to move from location A to location B.

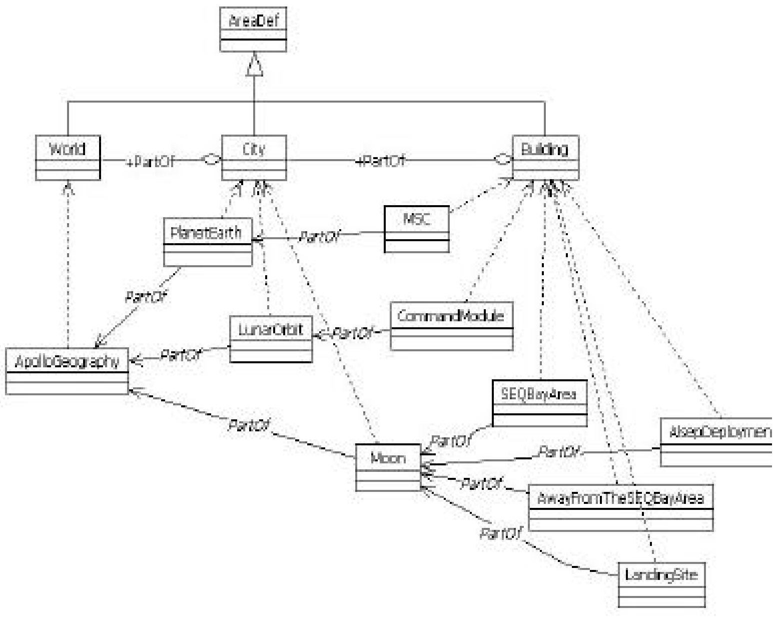
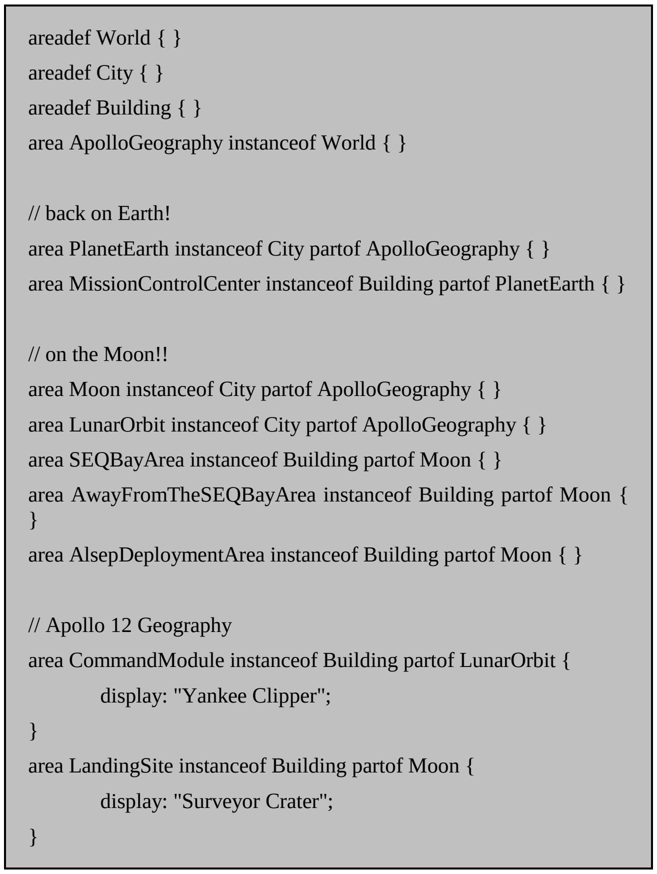


Figure 7. Apollo Geography Model Design



```
areadef World { }
areadef City { }
areadef Building { }
area ApolloGeography instanceof World { }

// back on Earth!
area PlanetEarth instanceof City partof ApolloGeography { }
area MissionControlCenter instanceof Building partof PlanetEarth { }

// on the Moon!!
area Moon instanceof City partof ApolloGeography { }
area LunarOrbit instanceof City partof ApolloGeography { }
area SEQBayArea instanceof Building partof Moon { }
area AwayFromTheSEQBayArea instanceof Building partof Moon {
}
area AlsepDeploymentArea instanceof Building partof Moon { }

// Apollo 12 Geography
area CommandModule instanceof Building partof LunarOrbit {
        display: "Yankee Clipper";
}
area LandingSite instanceof Building partof Moon {
        display: "Surveyor Crater";
}
```

Figure 8. Geography Model Brahms Source Code

2 During the Apollo days the NASA center in Houston was called the Manned Spaceflight Center (MSC). Today it is referred to as Johnson Space Center (JSC).

3 We are currently re-implementing the engine in Java.

8.3 Detecting Agents and Objects

As both agents arrive at their new location area they will immediately detect facts about the location of other agents and objects that are also in the area they arrive at. The simulation engine automatically creates beliefs for the agent from the facts about other objects and agents that are in the same location. The agents already in that location will get the belief that the agent that arrived is now also in the location. This way, agents will always notice other agents and objects that are in the location the same area.

9. Activity Model

In this section we describe the *Open SEQ Bay Door* activity performed during the ALSEP Offload. This model represents a part of the work practice of the Apollo 12 lunar surface astronauts as they performed the ALSEP Offload activity.

There are three separate areas where people are located during the Apollo ALSEP Offload activity. The CapCom is located at MSC. His main function is to listen to and communicate directly over the voice-loop with the astronauts. The CDR and LMP are the astronauts on the lunar surface and are located at or near the area of the SEQ Bay, which is located on the backside of the Lunar Module (LM) "Intrepid". The CMP is orbiting around the moon in the Command Module (CM) "Yankee Clipper."

9.1 Open SEQ Bay Door Activity

The ALSEP Offload starts at 116 hours 31 minutes and 34 seconds ground-elapsed time (GET)4, with the LMP announcing that they're starting the offload of the ALSEP. The next activity is for the LMP to open the SEQ Bay door. We describe how we modeled this activity in Brahms as a multi-agent model, based on the available Lunar Surface Journal data [24].

There are three high-level (sub)activities that one can identify in this OpenSEQBayDoor activity. First, there is a communication to MSC in Houston that they are ready to offload the ALSEP. This is the communication starting at 116:31:34. The issue to solve for the modeler is when this activity ends and the next activity begins. From the CDR communication at 116:32:02 we can infer that this is the time that the LMP actually opens the SEQ Bay door by pulling at the SEQ Bay door lanyard ribbons. So, we could start the activity of raising the SEQ Bay door around that time. However, from the video of the Apollo 145 ALSEP Offload it can be shown that before the LMP can pull the lanyard ribbons he has to grab them from the SEQ Bay, walk back until the ribbons are tight, and only then pull the ribbons to raise the SEQ Bay door. These activities have to happen before 116:32:02.

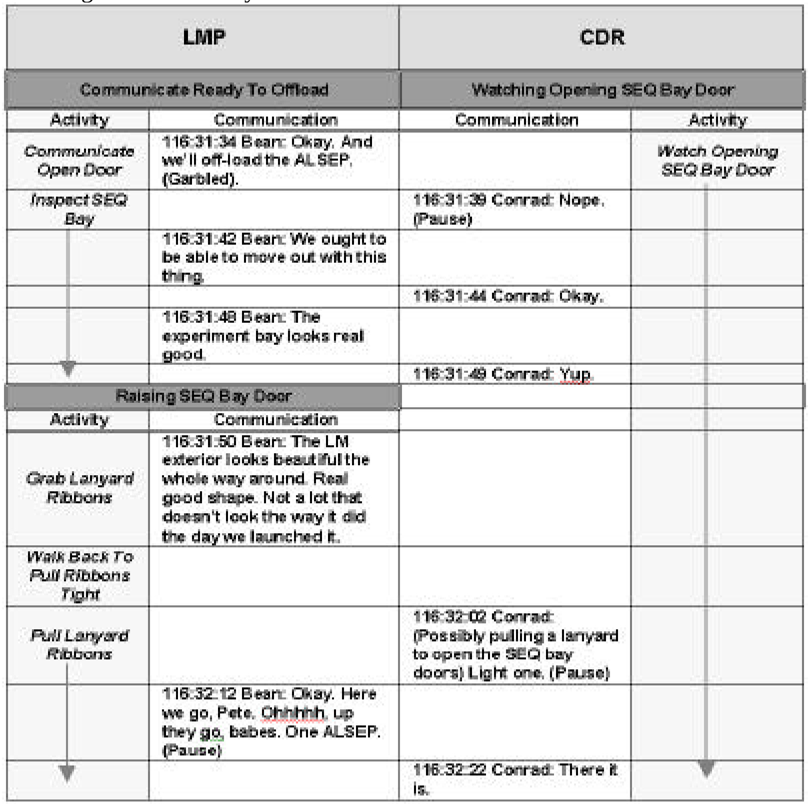
Table 1 shows the activities and sub-activities of the *Open SEQ Bay Door* activity for both LMP and CDR, mapped onto the communication transcribed in the Apollo LSJ. The actual names of the activities and sub-activities are more or less arbitrary, and conceptualize the modeler's *interpretation* of the observations of the Apollo 12 communication data and the Apollo 14 video data. However, these data and observations are strong evidence that these are the actual activities that are performed during the Open SEQ Bay Door activity.

The activities from Table 1 are implemented in the Brahms model as the *OpenSEQBayDoor* composite-activity. Figure 9 shows this activity, its sub-activities and workframes. Each (sub)activity is "executed" by a workframe, which means that when an agent executes the workframe the activity is performed within the context of that workframe.

An agent has an individual set of workframes inherited from the groups it belongs to. A workframe is a *production-rule* with preconditions matching the beliefs of an agent. When the preconditions of a workframe match with beliefs of the agent it becomes available. The simulation engine schedules the next activity of an agent based on her set of current, available and interrupted workframes.

As the first activity during the ALSEP offload, the CDR and LMP start walking to the area of the SEQ Bay. Walking to the SEQ Bay area to start opening the SEQ Bay door is modeled by the Move activity, as can be seen at the top of Figure 9. Now that we defined the sub-activities and workframes of the OpenSeqBayDoor activity the question is; how does the CDR and LMP agent start this activity during the simulation? Figure 10 shows the workframes of the AlsepOffload activity that both agents can execute to offload the ALSEP.

The first workframe to fire—the highest-level workframe, but lowest in Figure 10—is the *OffloadingAlsep* workframe, which executes the *AlsepOffload* activity. Executing the *AlsepOffload* activity enables all the workframes in it to potentially fire for the agent. Each of the workframes will execute lower-level activities, which are said to be subsumed by the higher-level activity.

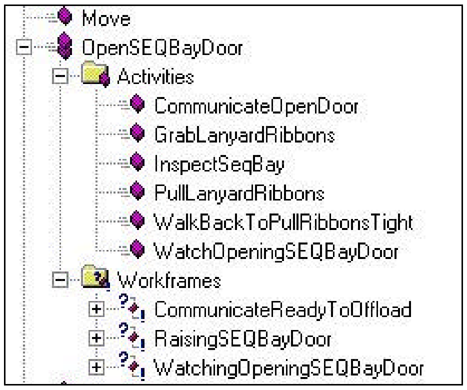


LMP		CDR	
Communicate Ready To Offload		Watching Opening SEQ Bay Door	
Activity	Communication	Communication	Activity
Communicate Open Door	116:31:34 Bean: Okay. And we'll off-load the ALSEP. (Garbled).		*Watch Opening SEQ Bay Door*
Inspect SEQ Bay		116:31:39 Conrad: Nope. (Pause)	
	116:31:42 Bean: We ought to be able to move out with this thing.		
		116:31:44 Conrad: Okay.	
	116:31:48 Bean: The experiment bay looks real good.		
		116:31:49 Conrad: Yup.	
Raising SEQ Bay Door			
Activity	Communication		
Grab Lanyard Ribbons	116:31:50 Bean: The LM exterior looks beautiful the whole way around. Real good shape. Not a lot that doesn't look the way it did the day we launched it.		
Walk Back To Pull Ribbons Tight			
Pull Lanyard Ribbons		116:32:02 Conrad: (Possibly pulling a lanyard to open the SEQ bay doors) Light one. (Pause)	
	116:32:12 Bean: Okay. Here we go, Pete. Ohhhhh, up they go, babes. One ALSEP. (Pause)		
		116:32:22 Conrad: There it is.	

Table 1 Open SEQ Bay Door Activity

4 The ground-elapsed time (GET) was the time clock in Houston that was started at the moment of launch.

5 Due to a malfunction of the television camera during the Apollo 12 mission, there is no video of the Apollo 12 ALSEP Offload activity.



Move
OpenSEQBayDoor
Activities
CommunicateOpenDoor
GrabLanyardRibbons
InspectSeqBay
PullLanyardRibbons
WalkBackToPullRibbonsTight
WatchOpeningSEQBayDoor
Workframes
CommunicateReadyToOffload
RaisingSEQBayDoor
WatchingOpeningSEQBayDoor

Figure 9 The OpenSEQ BayDoor composite-activity, sub-activities, and workframes

We can represent the relationship between workframes executing activities, containing other workframes that execute activities, etc, in a workframe-activitiy subsumption hierachy as shown in Figure 11.

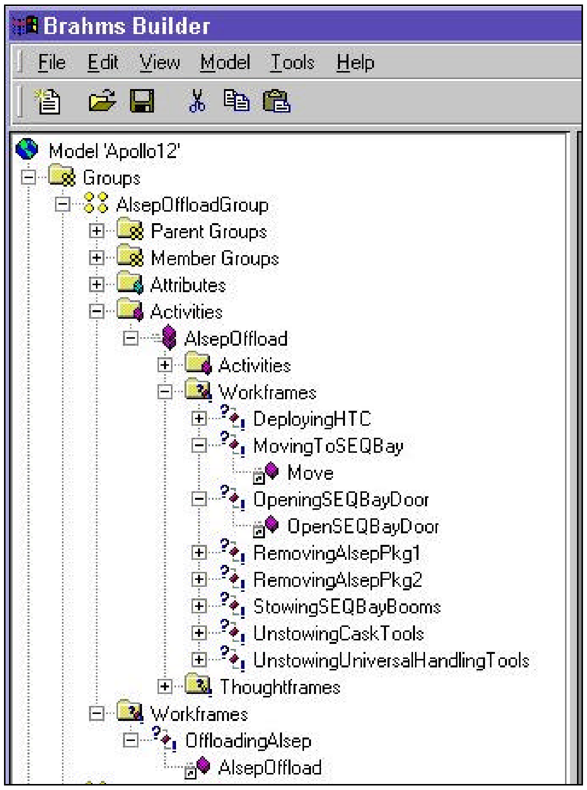
Only one activity can be active at any given time (i.e. at any clock-tick), consequently only one workframe is "being worked on" at any given time. This means that the order in which workframes at the same level in the hierarchy fire depends on two things; first, the conditions of the workframe that are to be matched to the beliefs of the agent, and second, the priority of the activities within the workframes.

There are a number of important language constructs (such as detectables, consequences, and thoughtframes) we are leaving out from the discussion in order to keep the length of the paper within the necessary limits. For a more detailed description of the Brahms language we refer the reader to [25].

9.2 Viewing the simulation results

Figure 12, shows the AlsepOffload activity performed by the AlBean and PeteConrad agents, as well as the communication between the two agents. While performing the AlsepOffload composite activity, both agents are within the OpenSEQBayDoor activity. While AlBean is performing the activities within the CommunicateReady- and the RaisingSEQBayDoor workframe, the PeteConrad agent is performing the activities within the WatchingOpenSEQBayDoor workframe. The grain-size of the simulation is one second. This means that the simulation engine updates every agent and object every second. We can therefore say that the simulation is a second by second model of the work practice of the lunar surface astronauts. Figure 12 also shows the location the agent when performing the activity. As an overlay, the (blue) dotted arrows show the communication of beliefs between agents AlBean and PeteConrad. The direction of the arrows show the direction in which the beliefs are being communicated, while the little square blue box at the start of the arrow shows the agent that is performing the communication.

Figure 12 is a screen shot from the AgentViewer application[6]. The AgentViewer application takes as input a Brahms Simulation History database[7]. This history database contains the historical situation-specific model data of a particular simulation run. The AgentViewer application creates a graphical representation of the activity of agents and objects during a simulation.



Brahms Builder
File Edit View Model Tools Help

Model 'Apollo12'
Groups
AlsepOffloadGroup
Parent Groups
Member Groups
Attributes
Activities
AlsepOffload
Activities
Workframes
DeployingHTC
MovingToSEQBay
Move
OpeningSEQBayDoor
OpenSEQBayDoor
RemovingAlsepPkg1
RemovingAlsepPkg2
StowingSEQBayBooms
UnstowingCaskTools
UnstowingUniversalHandlingTools
Thoughtframes
Workframes
OffloadingAlsep
AlsepOffload

Figure 10 The AlsepOffload workframes

10. Discussion and Future work

At this time we have used Brahms primarily as a simulation environment for simulating collaboration between people and systems. We have modeled a number of different organizations and work practice domains. The current Brahms language has been stable for the last year, and has allowed us to represent the way work actually happens for several different types of work practice (e.g. co-located and distributed office work, extra-vehicular deployment of instruments, collaboration between people and robots).

Currently, we are re-designing the simulation engine in Java. Next, we will add a Java-activity type to the language. Using this type of activity it will be possible to have a Brahms agent or object interface to and be integrated in other systems. This will create the ability to use Brahms not only as a simulation environment for understanding work processes and practice, but also as an environment for developing intelligent software agents that are based on the expertise and work practice of their human counter parts. This will make it possible to develop personal agent applications that include the intelligence and understanding of how the user works. We believe that this will enhance our ability to develop more human-centered systems. By developing agents that are more aware of their context and interactions with other agents and even their human users, we will be able to develop CSCW applications,

6 The AgentViewer application is a stand-alone Visual Basic application we developed for viewing the results of a simulation.

7 The history database is a complex relational database containing the simulation data preserving their relationships.

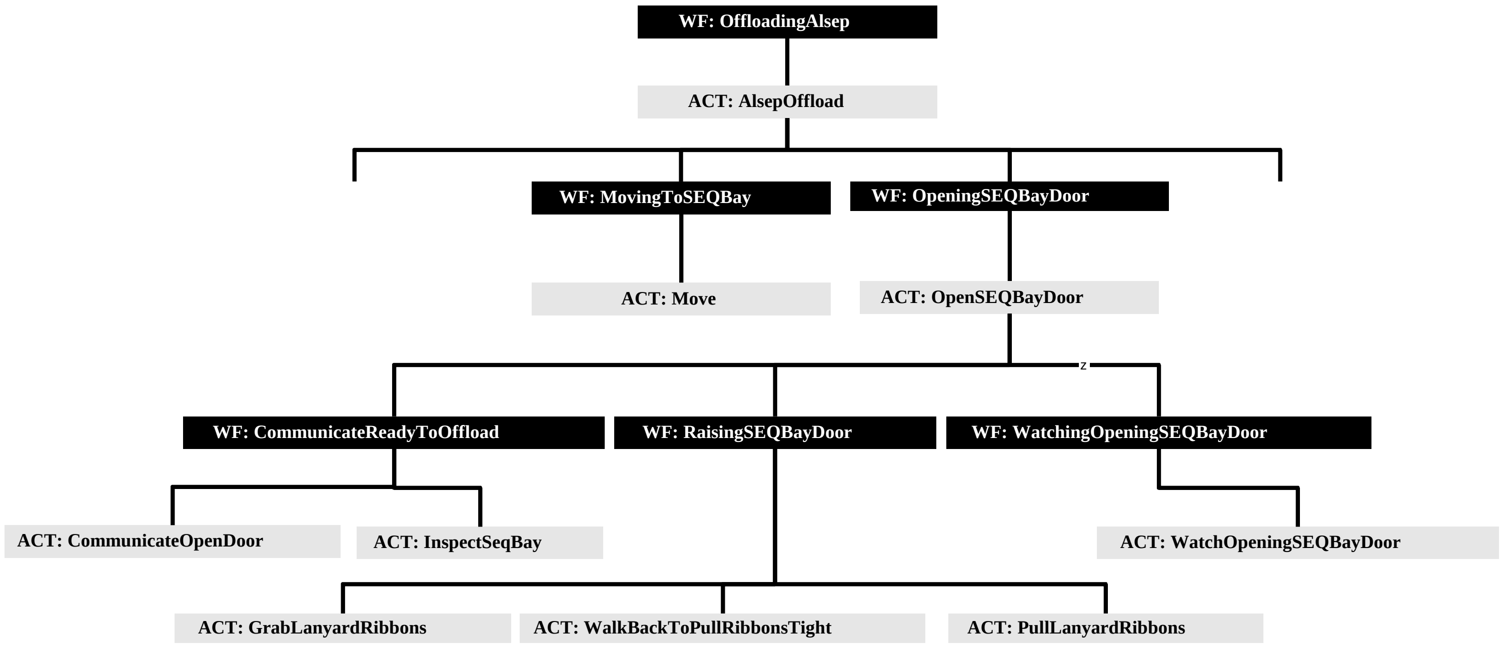


Figure 11 AlsepOffload Workframe-Activity Subsumption Hierarchy

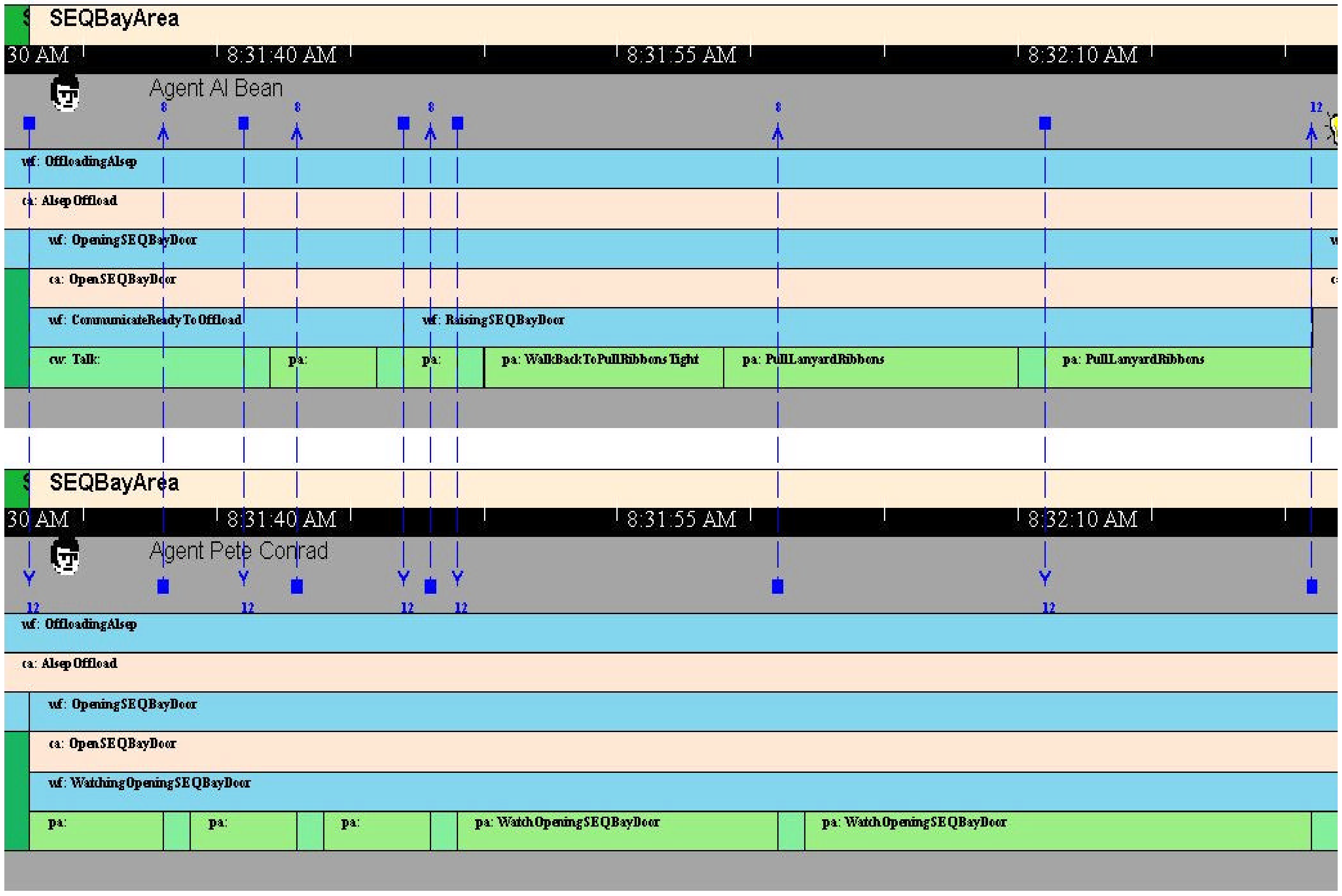


Figure 12. AlsepOffload Activity Agent Timeline

including decision-support and workflow systems that are more user-friendly and able to react to users according to the way the user works.

The risk of having agents perform unauthorized activities can be better controlled, because the developer of the agent can base the agent's behavior on the actual human behavior in the real world. This will allow us to develop better integration between the human work practice and the intelligent agent's work practice.

11. References

[1] W. J. Clancey, P. Sachs, M. Sierhuis, and R. van Hoof, "Brahms: Simulating practice for work systems design," *International Journal on Human-Computer Studies*, vol. 49, pp. 831-865, 1998.

[2] W. D. Compton, *Where No Man Has Gon Before; A History of Apollo Lunar Exploration Missions*, vol. NASA SP-4214. Washingotn, DC.: NASA Office of Management Scientific and Technical Information Division, 1989.

[3] D. E. Wilhelms, *To a Rocky Moon: A Geologist's History of Lunar Exploration*. Tucson: The University of Arizona Press, 1993.

[4] A. Chaikin, *A man on the moon: Voyages of the Apollo astronauts*. New York, NY: Penguin Books, 1994.

[5] R. Godwin, "Apollo 12 - The NASA Mission Reports," in *The NASA Mission Reports*, R. Godwin, Ed. Burlinton, Canada: Apogee Books, 1999.

[6] J. S. C. NASA, "Apollo Lunar Surface EVA Video," . Houston, TX: NASA/Johnson Space Center, Information Science Branch Video Repository (GP28).

[7] L. S. Vygotsky, *Mind in Society: The Development of Higher Psychological Processes*. Cambridge, MA: Harvard University Press, 1978, Originally published in Rusian in 1934.

[8] H. Nakashima, L. Noda, and K. Handa, "Organic Programming language GAEA for multi-agents," presented at Proceedings of the Second International Conference on Multi-Agent Systems, Kyoto, Japan, 1996.

[9] R. Brooks, A., "Intelligence without representation," *Artificial Intelligence*, vol. 47, pp. 139-159, 1991.

[10] W. J. Clancey, "The Conceptual Nature of Knowledge, Situations, and Activity," in *Human and Machine Expertise in Context*, P. Feltovich, R. Hoffman, and K. Ford, Eds. Menlo Park, CA: The AAAI Press, 1997, pp. 247-291.

[11] J. Greenbaum and M. Kyng, "Design at Work: Cooperative design of computer systems," . Hillsdale, NJ.: Lawrence Erlbaum, 1991.

[12] P. E. Agre, "Computational research on interaction and agency," *Artificial Intelligence*, vol. 72, pp. 1-52, 1995.

[13] D. Kirsh, "The intelligent use of space," *Artificial Intelligence*, vol. 73, pp. 31-68, 1995.

[14] Y. Lespérance and H. J. Levesque, "Indexical knowledge and robot action: A logical account," *Artificial Intelligence*, vol. 73, pp. 69-115, 1995.

[15] J. Searle, R., *Speech Acts*. Cambridge, UK: Cambridge University Press, 1969.

[16] J. Searle, R., "A taxonomy of illocutionary acts," in *Language, Mind, and Knowledge*, vol. 1-29, K. Gunderson, Ed. Minneapolis: University of Minnesota, 1975, pp. 344-369.

[17] E. Wenger, *Communities of Practice; Learning, meaning, and identity*: Cambridge University Press, 1997.

[18] K. Konolige, "A first-order formalization of knowledge and action for a multi-agent planning system," in *Machine Intelligence*, vol. 10, J. E. Hayes, D. Mitchie, and Y. Pao, Eds. Chichester, England: Ellis Horwood, 1982, pp. 41-72.

[19] K. Konolige, *A Deduction Model of Belief*. San Mateo, CA: Morgan Kaufmann, 1986.

[20] J. Hintikka, *Knowledge and Belief*. Ithaca, NY: Cornell University Press, 1962.

[21] R. C. Moore, "A formal theory of knowledge and action," in *Readings in Planning*, J. F. Allen, J. Hendler, and A. Tate, Eds. San Mateo, CA: Morgan Kaufmann Publishers, 1990, pp. 480-519.

[22] L. A. Suchman, *Plans and Situated Action: The Problem of Human Machine Communication*. Cambridge, MA: Cambridge University Press, 1987.

[23] J. Rumbaugh, I. Jacobson, and G. Booch, *The Unified Modeling Language Reference Manual*: Addison-Wesley, 1998.

[24] E. Jones, M., "The Apollo Lunar Surface Journal," National Aeronautics and Space Administration, WWW URL: http://www.hq.nasa.gov/office/pao/History/alsj/, 1996 1997.

[25] R. van Hoof and M. Sierhuis, "Brahms Language Reference," : http://www.agentisolutions.com, 2000.